

## IN THE CLAIMS:

Please cancel claims 16 and 19, and amend the claims as follows:

Claim 1 (Currently Amended): Method for scheduling the service of a thread, said method comprising the steps of:

masking interrupts from one or more hardware devices in order to ignore interrupts for other threads;

acquiring ~~receiving~~ a latency information associated with the thread;

unmasking interrupts from the one or more hardware devices in order to detect interrupts for the other threads; and

rearranging an order in which the thread and the other threads will be serviced to schedule ~~scheduling~~ the thread for processing in accordance with said latency information.

Claim 2 (Currently Amended): The method of claim 1, wherein said latency information is computed based on a buffer size or display rate associated with an interrupt and said scheduled thread is for servicing said interrupt.

Claim 3 (Currently Amended): The method of claim 2, wherein said latency information is representative of [[real]] time units.

Claim 4 (Original): The method of claim 3, wherein said latency information represents a time duration that is necessary to service the thread.

Claim 5 (Original): The method of claim 3, wherein said latency information represents a time at which said scheduled thread will be processed.

Claim 6 (Original): The method of claim 3, wherein said latency information represents a time duration that is necessary to setup the thread.

Claim 7 (Currently Amended): The method of claim 3, wherein said latency information is dependent on a hardware constraint for one of the one or more hardware devices.

Claim 8 (Original): The method of claim 3, wherein said latency information is provided by a device driver.

Claim 9 (Currently Amended): Apparatus for scheduling the service of a thread, said apparatus comprising:

means for receiving an interrupt from a hardware device;

means for masking interrupts from one or more hardware devices in order to ignore interrupts for other threads;

means for acquiring latency information associated with the interrupt;

means for unmasking interrupts from the one or more hardware devices in order to detect interrupts for the other threads; and

means for rearranging an order in which the thread and the other threads will be serviced to schedule scheduling the thread to process the interrupt in accordance with said latency information.

Claim 10 (Currently Amended): The apparatus of claim 9, wherein said latency information is representative of [[real]] time units.

Claim 11 (Currently Amended): The apparatus of claim 9, wherein said latency information is dependent on a hardware constraint for one of the one or more hardware devices.

Claim 12 (Original): The apparatus of claim 11, wherein said hardware constraint is a size of a buffer.

Claim 13 (Original): The apparatus of claim 11, wherein said hardware constraint is a fullness of a buffer.

Claim 14 (Currently Amended): The apparatus of claim 11, wherein said hardware constraint is dynamically computed based on a buffer size or display rate defined.

Claim 15 (Original): The apparatus of claim 9, wherein said latency information is generated by a device driver associated with the hardware device.

Claim 16 (Canceled)

Claim 17 (Currently Amended): The method of claim 1[[6]], further comprising toggling an interrupt line.

Claim 18 (Currently Amended): The method of claim 1[[6]], further comprising:  
determining the thread should be activated; and  
activating the thread for processing.

Claim 19 (Canceled)

Claim 20 (Currently Amended): The method of claim 1[[6]], wherein said latency information [[value]] represents a time duration that is used to determine when the thread should be activated for processing.

Claim 21 (New): The method of claim 1, further comprising:  
creating the thread for interrupt processing when one of the one or more hardware devices is initialized; and  
freeing the thread when the one of the one or more hardware devices is shut down.

Claim 22 (New): The method of claim 21, further comprising:  
creating an additional thread for interrupt processing of another interrupt that the one of the one or more hardware devices is configured to generate, wherein a first interrupt identification number is associated with the thread and a second interrupt

identification number that is different than the first interrupt identification number is associated with the additional thread; and

freeing the additional thread when the one of the one or more hardware devices is shut down.